

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
KHOA AN TOÀN THÔNG TIN**



**BÁO CÁO BÀI THỰC HÀNH
HỌC PHẦN: THỰC TẬP CƠ SỞ
MÃ HỌC PHẦN: INT13147**

**BÀI THỰC HÀNH 4.2
LẬP TRÌNH THUẬT TOÁN MẬT MÃ HỌC**

Sinh viên thực hiện:

B22DCAT253 Đinh Thị Thanh Tâm

Giảng viên hướng dẫn: TS Đinh Trường Duy

HỌC KỲ 2 NĂM HỌC 2024-2025

MỤC LỤC

MỤC LỤC	2
DANH MỤC CÁC HÌNH VẼ.....	3
CHƯƠNG 1. GIỚI THIỆU CHUNG VỀ BÀI THỰC HÀNH	4
1.1 Mục đích.....	4
1.2 Tìm hiểu lý thuyết	4
1.2.1 Tìm hiểu về lập trình số lớn với các phép toán cơ bản	4
1.2.2 Tìm hiểu về giải thuật mật mã khóa công khai RSA	5
CHƯƠNG 2. NỘI DUNG THỰC HÀNH	7
2.1 Chuẩn bị môi trường	7
2.2 Các bước thực hiện.....	7
kết luận	11
TÀI LIỆU THAM KHẢO	11

DANH MỤC CÁC HÌNH VẼ

Hình 1 Hàm kiểm tra số nguyên tố và kết quả thử nghiệm với số lớn.....	7
Hình 2 Tìm UCLN và kết quả thử nghiệm với số lớn.....	8
Hình 3 Tìm nghịch đảo modular: $d \equiv e^{-1} \pmod{\phi}$ và kết quả thử nghiệm với số lớn.....	8
Hình 4 Hàm sinh số nguyên tố ngẫu nhiên trong khoảng	8
Hình 5 Thuật toán tạo khóa RSA	9
Hình 6 Hàm mã hóa giải mã.....	9
Hình 7 Chương trình chính	10
Hình 8 Nhập thông điệp từ bàn phím	10
Hình 9 Kết quả mã hóa và giải mã thông điệp bằng thuật toán RSA.....	10

CHƯƠNG 1. GIỚI THIỆU CHUNG VỀ BÀI THỰC HÀNH

1.1 Mục đích

Sinh viên tìm hiểu một giải thuật mã hóa phổ biến và lập trình được chương trình mã hóa và giải mã sử dụng ngôn ngữ lập trình phổ biến như C/C++/Python/Java, đáp ứng chạy được với số lớn.

1.2 Tìm hiểu lý thuyết

1.2.1 Tìm hiểu về lập trình số lớn với các phép toán cơ bản

Một số lý thuyết cơ bản về lập trình số nguyên lớn:

- Biểu diễn số nguyên lớn: Trong lập trình số nguyên lớn, số này thường được biểu diễn dưới dạng mảng hoặc danh sách các chữ số. Mỗi phần tử trong mảng hoặc danh sách này thường lưu trữ một chữ số của số nguyên.
- Cộng và Trừ:
 - Các phép toán cộng và trừ có thể được thực hiện bằng cách duyệt qua từng chữ số của hai số và thực hiện phép toán tương ứng từng cặp chữ số, kèm theo việc xử lý nhớ và mượn nếu cần.
 - Bắt đầu từ chữ số đơn vị và tiếp tục lần lượt đến chữ số hàng đơn vị, thực hiện phép toán cộng hoặc trừ tương ứng.
- Nhân:
 - Phép nhân hai số nguyên lớn có thể thực hiện bằng thuật toán nhân dài (long multiplication) hoặc các thuật toán nhân nhanh hơn như nhân Karatsuba hoặc nhân Toom-Cook.
 - Thuật toán nhân dài thực hiện phép nhân tương tự như cách thực hiện phép nhân trong toán học cơ bản: duyệt qua từng chữ số của một số và nhân với từng chữ số của số kia, sau đó cộng các kết quả lại.
- Chia:
 - Phép chia hai số nguyên lớn có thể thực hiện bằng thuật toán chia dài (long division) hoặc các thuật toán chia hiệu quả hơn như thuật toán chia như thuật toán chia nhân bán (divide-and-conquer division) hoặc thuật toán chia nhưng tốt hơn (Newton-Raphson division).
 - Tương tự như phép nhân, thuật toán chia dài duyệt qua từng chữ số của số chia và thực hiện phép chia tương ứng.
- Tính Lũy Thừa:
 - Phép tính lũy thừa của một số nguyên lớn có thể được thực hiện bằng thuật toán lũy thừa nhanh (exponentiation by squaring). Thuật toán này giảm thiểu số lượng phép toán bằng cách phân tích số mũ thành các phần nhỏ và tính lũy thừa của từng phần.

- Tìm Ước Chung Lớn Nhất (GCD):
 - Phép tìm ước chung lớn nhất của hai số nguyên lớn có thể được thực hiện bằng thuật toán Euclid mở rộng (extended Euclidean algorithm) hoặc thuật toán nhị phân (binary GCD algorithm). Cả hai thuật toán này đều dựa trên việc lặp đi lặp lại một số phép chia và tính toán dựa trên phần dư.

Trong lập trình số nguyên lớn, việc tối ưu hóa và cải thiện hiệu suất là rất quan trọng. Điều này bao gồm việc sử dụng các thuật toán hiệu quả nhất để thực hiện các phép toán cơ bản và giải quyết các vấn đề liên quan đến hiệu năng và bộ nhớ.

Trong Python, số nguyên không bị giới hạn về giá trị. Tuy nhiên, số nguyên có thể lớn đến mức nào phụ thuộc vào bộ nhớ có sẵn trên hệ thống mà bạn đang sử dụng. Chính vì vậy, ta có thể xử lý số nguyên lớn dễ dàng với Python

1.2.2 Tìm hiểu về giải thuật mật mã khóa công khai RSA

1.2.2.1 Giải thuật mã hóa khóa công khai

Mã hóa khóa bất đối xứng (Hay còn gọi là mã hóa khóa công khai): là phương pháp mã hóa mà khóa sử dụng để mã hóa sẽ khác với khóa dùng để giải mã. Khóa dùng để mã hóa gọi là khóa công khai và khóa dùng để giải mã gọi là khóa bí mật. Tất cả mọi người đều có thể biết được khóa công khai (kể cả hacker), và có thể dùng khóa công khai để mã hóa thông tin. Nhưng chỉ có người nhận mới nắm giữ khóa bí mật, nên chỉ có người nhận mới có thể giải mã được thông tin.

Đặc điểm nổi bật của các hệ mã hóa khóa bất đối xứng là kích thước khóa lớn, lên đến hàng ngàn bit. Do vậy, các hệ mã hóa dạng này thường có tốc độ thực thi chậm hơn nhiều lần so với các hệ mã hóa khóa đối xứng với độ an toàn tương đương. Mặc dù vậy, các hệ mã hóa khóa bất đối xứng có khả năng đạt độ an toàn cao và ưu điểm nổi bật nhất là việc quản lý và phân phối khóa đơn giản hơn do khóa công khai có thể phân phối rộng rãi.

Hệ thống mật mã hóa khóa công khai có thể sử dụng với các mục đích:

- Mã hóa: giữ bí mật thông tin và chỉ có người có khóa bí mật mới giải mã được.
- Tạo chữ ký số: cho phép kiểm tra văn bản có phải đã được tạo với một khóa bí mật nào đó hay không.
- Thỏa thuận khóa: cho phép thiết lập khóa dùng để trao đổi thông tin giữa hai bên.

Các giải thuật mã hóa khóa bất đối xứng điển hình bao gồm: RSA, Rabin, ElGamal, McEliece và Knapsack. Trong bài tiểu luận này, chúng tôi tìm hiểu và trình bày về giải thuật mã hóa RSA – một trong các giải thuật mã hóa khóa đối xứng được sử dụng rộng rãi nhất trên thực tế.

1.2.2.2 Giải thuật mã hóa khóa công khai RSA

Thuật toán mã hóa RSA là một thuật toán mã hóa khóa công khai. Thuật toán RSA được xây dựng bởi các tác giả Ron Rivest, Adi Shamir và Len Adleman tại học viện MIT vào năm 1977, và ngày nay đang được sử dụng rộng rãi. Về mặt tổng quát thuật toán RSA là một phương pháp mã hóa theo khối. Trong đó thông điệp M và bản mã C là các số nguyên

từ 0 đến 2^i với i số bit của khối. Kích thước thường dùng của i là 1024 bit. Thuật toán RSA sử dụng hàm một chiều là vấn đề phân tích một số thành thừa số nguyên tố và bài toán Logarith rời rạc.

Để thực hiện mã hóa và giải mã, thuật toán RSA dùng phép lũy thừa modulo của lý thuyết số.

- Quá trình sinh khóa
 - Chọn hai số nguyên tố lớn p và q và tính $N = pq$. Cần chọn p và q sao cho: $M < 2^{i-1} < N < 2^i$ với i là chiều dài bản rõ.
 - Tính $\Phi(n) = (p - 1)(q - 1)$
 - Tìm một số e sao cho: $\{e \text{ và } \Phi(n) \text{ là 2 số nguyên tố cùng nhau và } 0 < e < \Phi(n)\}$
 - Tìm một số d sao cho: $e \cdot d \equiv 1 \pmod{\Phi(n)}$ (hay: $d = e^{-1} \pmod{\Phi(n)}$) (d là nghịch đảo của e trong phép modulo $\Phi(n)$)
 - Chọn khóa công khai K_u là cặp (e, N) , khóa riêng K_r là cặp (d, N)
- Quá trình mã hóa:
 - Thông điệp m được chuyển thành số: $m < n$
 - Bản mã $c = m^e \pmod{n}$
- Quá trình giải mã:
 - Bản mã c , $c < n$
 - Bản rõ $m = c^d \pmod{n}$
- Lưu ý: Khi chọn hai số nguyên tố p và q để sử dụng trong hệ mã hoá RSA, quan trọng nhất là chúng không nên quá gần nhau hoặc quá lệch nhau.

CHƯƠNG 2. NỘI DUNG THỰC HÀNH

2.1 Chuẩn bị môi trường

- Sử dụng ngôn ngữ lập trình Python

2.2 Các bước thực hiện

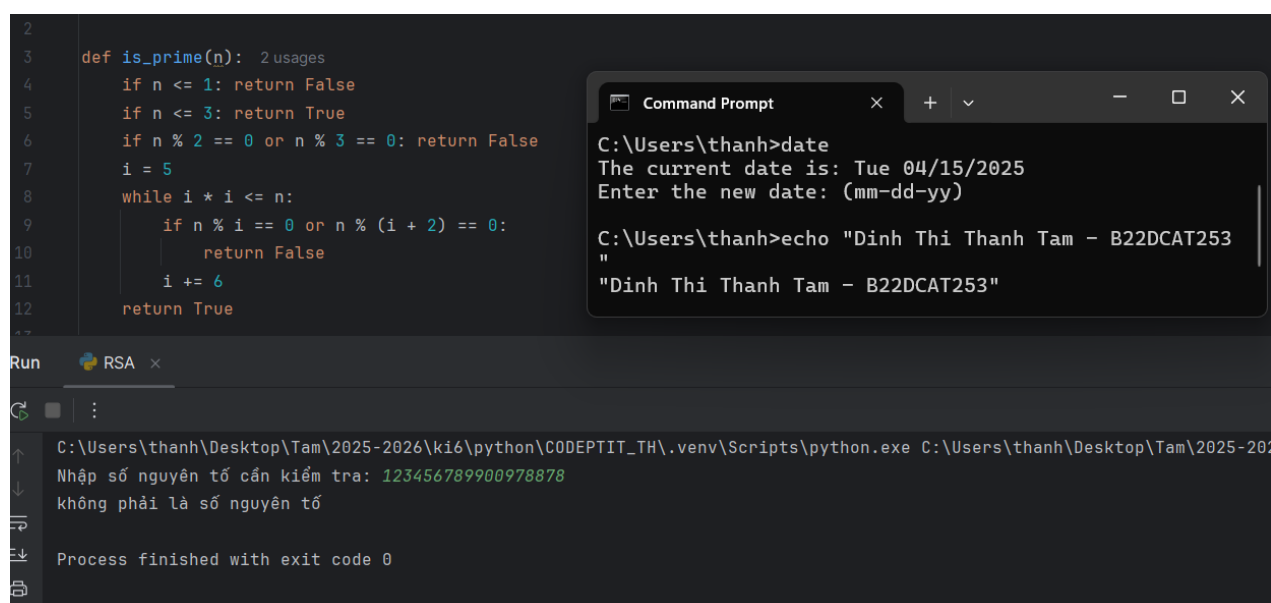
1. Lập trình thư viện số lớn với các phép toán cơ bản để sử dụng trong giải thuật mã hóa/giải mã RSA
2. Lập trình thư viện số lớn với các phép toán cơ bản để sử dụng trong giải thuật mã hóa/giải mã RSA
3. Thử nghiệm chứng minh thư viện hoạt động tốt với các ví dụ phép toán cho số lớn
4. Lập trình giải thuật mã hóa và giải mã
5. Thử nghiệm mã hóa và giải mã chuỗi ký tự: "I am <mã sinh viên>" (thay bằng mã sinh viên: B22DCAT253)

- `is_prime(n)`

Chức năng: Kiểm tra xem n có phải là số nguyên tố không.

Nguyên lý: Duyệt từ 2 đến căn n , kiểm tra nếu chia hết thì không phải nguyên tố.

Dùng để: Tạo các số nguyên tố p và q cho RSA.



```
2
3 def is_prime(n): 2 usages
4     if n <= 1: return False
5     if n <= 3: return True
6     if n % 2 == 0 or n % 3 == 0: return False
7     i = 5
8     while i * i <= n:
9         if n % i == 0 or n % (i + 2) == 0:
10             return False
11         i += 6
12     return True
13
Run RSA x
```

Command Prompt

```
C:\Users\thanh>date
The current date is: Tue 04/15/2025
Enter the new date: (mm-dd-yy)

C:\Users\thanh>echo "Đinh Thi Thanh Tam - B22DCAT253"
"Đinh Thi Thanh Tam - B22DCAT253"
```

C:\Users\thanh\Desktop\Tam\2025-2026\ki6\python\CODEPTIT_TH\.venv\Scripts\python.exe C:\Users\thanh\Desktop\Tam\2025-2026\ki6\python\CODEPTIT_TH\main.py

Nhập số nguyên tố cần kiểm tra: 123456789900978878

không phải là số nguyên tố

Process finished with exit code 0

Hình 1 Hàm kiểm tra số nguyên tố và kết quả thử nghiệm với số lớn

- `gcd(a, b)`

Chức năng: Tính ước chung lớn nhất của a và b bằng thuật toán Euclid.

Dùng để: Đảm bảo e và $\phi(n)$ là nguyên tố cùng nhau ($\text{gcd} = 1$).

```

13
14 def gcd(a, b):
15     while b != 0:
16         a, b = b, a % b
17     return a

Run RSA x

C:\Users\thanh>date
The current date is: Tue 04/15/2025
Enter the new date: (mm-dd-yy)

C:\Users\thanh>echo "Dinh Thi Thanh Tam - B22DCAT253"
"Dinh Thi Thanh Tam - B22DCAT253"

C:\Users\thanh\Desktop\Tam\2025-2026\ki6\python\CODEPTIT_TH\.venv\Scripts\python.exe C:\Users\thanh\Desktop\Tam\2025-2026\ki6\python\CODEPTIT_TH\rsa.py
nhập số thứ nhất: 12333333333333333333
nhập số thứ hai: 34555555555555555555
UCLB của hai số là 1

```

Hình 2 Tìm UCLN và kết quả thử nghiệm với số lớn

- `modinv(e, phi)`

Chức năng: Tính nghịch đảo modular của e theo modulo phi.

Tức là: tìm d sao cho $(d * e) \% \phi == 1$

Dùng thuật toán: Euclid mở rộng.

Dùng để: Tính khóa riêng (private key) d.

```

19 def modinv(e, phi):
20     def egcd(a, b):
21         return g, x - (b // a) * y, y
22     g, x, y = egcd(e, phi)
23     if g != 1:
24         raise Exception('Không tìm được nghịch đảo modular')
25     else:
26         return x % phi

Run RSA x

C:\Users\thanh>date
The current date is: Tue 04/15/2025
Enter the new date: (mm-dd-yy)

C:\Users\thanh>echo "Dinh Thi Thanh Tam - B22DCAT253"
"Dinh Thi Thanh Tam - B22DCAT253"

C:\Users\thanh\Desktop\Tam\2025-2026\ki6\python\CODEPTIT_TH\.venv\Scripts\python.exe C:\Users\thanh\Desktop\Tam\2025-2026\ki6\python\CODEPTIT_TH\rsa.py
nhập số e: 65537
nhập số phi: 5444
nghịch đảo modular d = e ^ -1 mod phi: 4897

```

Hình 3 Tìm nghịch đảo modular: $d \equiv e^{-1} \bmod \phi$ và kết quả thử nghiệm với số lớn

- `generate_prime_candidate(start, end)`

Chức năng: Sinh ngẫu nhiên một số nguyên tố trong khoảng cho trước.

Dùng để: Sinh hai số nguyên tố p và q.

```

def generate_prime_candidate(start=100, end=300):
    while True:
        num = random.randint(start, end)
        if is_prime(num):
            return num

C:\Users\thanh>date
The current date is: Tue 04/15/2025
Enter the new date: (mm-dd-yy)

C:\Users\thanh>echo "Dinh Thi Thanh Tam - B22DCAT253"
"Dinh Thi Thanh Tam - B22DCAT253"

```

Hình 4 Hàm sinh số nguyên tố ngẫu nhiên trong khoảng

- `generate_keypair()`

Chức năng: Tạo cặp khóa RSA:

- Sinh p, q , tính $n = p \cdot q$
- Tính $\phi(n)$
- Chọn số e sao cho nguyên tố cùng nhau với $\phi(n)$
- Tính $d = e^{-1} \bmod \phi(n)$
- Trả về: $(\text{public_key}, \text{private_key})$

The image shows a Python IDE with a function `generate_keypair()` defined. The function generates two prime numbers p and q , calculates $n = p \cdot q$ and $\phi(n) = (p-1) \cdot (q-1)$. It then selects a random value e between 2 and $\phi(n)$ such that $\gcd(e, \phi(n)) = 1$, and calculates the modular inverse d of e modulo $\phi(n)$. Finally, it returns the public key (e, n) and the private key (d, n) .

Overlaid on the IDE is a Windows Command Prompt window. It shows the user running `date` (displaying 'Tue 04/15/2025'), then `echo "Dinh Thi Thanh Tam - B22DCAT253"`, which outputs the string "Dinh Thi Thanh Tam - B22DCAT253".

Hình 5 Thuật toán tạo khóa RSA

▪ `encrypt(pk, plaintext)`

Chức năng: Mã hóa chuỗi plaintext bằng khóa pk (public)

Cách làm:

- Chuyển từng ký tự thành số (`ord(char)`)
- Mã hóa từng số: $C = M^e \bmod n$
- Trả về: danh sách số nguyên là bản mã.

▪ `decrypt(pk, ciphertext)`

Chức năng: Giải mã danh sách số nguyên ciphertext bằng khóa pk (private).

Cách làm:

- Từng số: $M = C^d \bmod n$
- Chuyển số về ký tự (`chr()`)
- Nối lại thành chuỗi gốc.

The image shows a Python IDE with two functions: `encrypt(pk, plaintext)` and `decrypt(pk, ciphertext)`. The `encrypt` function iterates over each character in the plaintext, converts it to its ASCII value, and calculates $C = M^e \bmod n$. The `decrypt` function iterates over each number in the ciphertext, calculates $M = C^d \bmod n$, and converts it back to a character. The results are joined back into a string.

Overlaid on the IDE is a Windows Command Prompt window, identical to the one in Figure 5, showing the current date and the echo of the string "Dinh Thi Thanh Tam - B22DCAT253".

Hình 6 Hàm mã hóa giải mã

```

59
60 # == Chương trình chính ==
61 print("Tạo khóa RSA...")
62 public, private = generate_keypair()
63 print("Public key:", public)
64 print("Private key:", private)
65
66 # Nhập thông điệp từ bàn phím

```

```

C:\Users\thanh>date
The current date is: Tue 04/15/2025
Enter the new date: (mm-dd-yy)

C:\Users\thanh>echo "Dinh Thi Thanh Tam - B22DCAT253"
"Dinh Thi Thanh Tam - B22DCAT253"

```

Hình 7 Chương trình chính

```

65
66 # Nhập thông điệp từ bàn phím
67 message = input("Nhập thông điệp cần mã hóa: ")
68
69 encrypted_msg = encrypt(public, message)
70 print("Thông điệp sau mã hóa:", encrypted_msg)
71
72 decrypted_msg = decrypt(private, encrypted_msg)
73 print("Thông điệp sau giải mã:", decrypted_msg)
74

```

```

C:\Users\thanh>date
The current date is: Tue 04/15/2025
Enter the new date: (mm-dd-yy)

C:\Users\thanh>echo "Dinh Thi Thanh Tam - B22DCAT253"
"Dinh Thi Thanh Tam - B22DCAT253"

```

Hình 8 Nhập thông điệp từ bàn phím

- Khi chạy chương trình, sẽ hiển thị dòng:
Nhập thông điệp cần mã hóa:
- Thử nghiệm mã hóa và giải mã chuỗi ký tự: “*I am B22DCAT253*”
- Chương trình sẽ hiển thị chuỗi đã mã hóa (dưới dạng danh sách số nguyên) và kết quả sau khi giải mã để kiểm tra tính chính xác.

```

Run RSA x
C:\Users\thanh\Desktop\Tam\2025-2026\ki6\python\CODEPTIT_TH\.venv\Scripts\python.exe C:\Users\thanh\Desktop\Tam\2025-2026\ki6\python\CODEPTIT_TH\main.py
Tạo khóa RSA...
Public key: (18583, 26969)
Private key: (5287, 26969)
Nhập thông điệp cần mã hóa: I am B22DCAT253
Thông điệp sau mã hóa: 128942473197835809247313652136981369813445141134598121313698625210347
Thông điệp sau giải mã: I am B22DCAT253
Process finished with exit code 0

```

```

C:\Users\thanh>date
The current date is: Tue 04/15/2025
Enter the new date: (mm-dd-yy)

C:\Users\thanh>echo "Dinh Thi Thanh Tam - B22DCAT253"
"Dinh Thi Thanh Tam - B22DCAT253"

```

Hình 9 Kết quả mã hóa và giải mã thông điệp bằng thuật toán RSA

KẾT LUẬN

- Lý thuyết về mã hóa khóa công khai
- Chạy giải thuật mã hóa RSA thành công

TÀI LIỆU THAM KHẢO

- [1] Đỗ Xuân Chợt, Bài giảng Mật mã học cơ sở, Học viện Công Nghệ Bưu Chính Viễn Thông, 2021.
- [2] Đỗ Xuân Chợt, Bài giảng Mật mã học nâng cao, Học viện Công Nghệ Bưu Chính Viễn Thông, 2021.